Seminar Nasional Teknologi Informasi & Komunikasi Terapan 2012 (Semantik 2012)
Semarang, 23 Juni 2012
ISBN 979 - 26 - 0255 - 0

APLIKASI STEGANOGRAFI UNTUK MENYEMBUNYIKAN TEKS DALAM MEDIA IMAGE DENGAN MENGGUNAKAN METODE LSB

Asep Saefullah[1], Himawan[2], Nazori Agani[3]

[1,2,3]*Mahasiswa Pasca Sarjana, Universitas Budi Luhur*
Jl. Ciledug Raya, Petukangan Utara, Jakarta Selatan, 12260
Email *asaefullah@gmail.com, himawanawan10@gmail.com, nazori@budiluhur.ac.id*

ABSTRAK

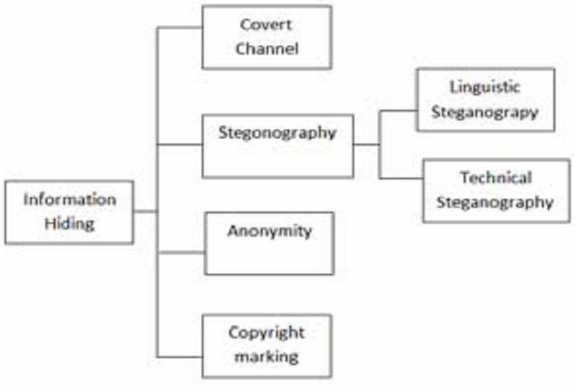
Data rahasia yang akan dikirim memerlukan perlidungan ekstra agar hanya dapat dibaca oleh target penerima saja. Untuk itu dirancang sebuah aplikasi steganografi yang bertujuan untuk menyamarkan eksistensi (keberadaan) data rahasia sehingga sulit dideteksi, dan melindungi hak cipta suatu produk. Metode yang dipergunakan pada tipe berkas citra adalah least significant bit (LSB), metode ini menyembunyikan data dengan mengganti bit-bit data yang paling tidak berarti di dalam cover dengan bit-bit data rahasia. Aplikasi steganografi yang dibuat terdiri dari 3 proses yaitu: meload image yang ingin ditambahkan pesan rahasia, menambahkan pesan ke dalam image (encode image) dan proses yang digunakan untuk menampilkan (extract) pesan rahasia yang ada dalam image. Ukuran semula pada image original dengan format .jpg akan mengalami kenaikan nilai setelah ditambahkan data rahasia dengan menggunakan format .png. Aplikasi steganografi ini berhasil menampilkan pesan rahasia yang ada dalam image, dengan tidak merubah cover citra image.

Kata kunci : Steganografi, LSB, Citra Image

1. PENDAHULUAN

Pengiriman data melalui media internet sudah menjadi kebutuhan karena sifatnya yang cepat, tepat dan mudah, namun demikian perlu menjadi perhatian aspek keamanan data pada saat pengiriman terutama data rahasia. Berkembangnya teknologi internet dan aplikasi menggunakan internet maka berkembang pula kejahatan sistem informasi. Dengan berbagai teknik banyak yang mencoba untuk mengakses informasi yang bukan haknya. Untuk melindungi data yang dikirim, dikembangkan berbagai teknik untuk melindungi informasi yang dirahasiakan dari orang yang tidak berhak, salah satunya adalah teknik steganografi. Steganografi sebagai suatu seni penyembunyian pesan ke dalam pesan lainnya yang telah ada sejak sebelum masehi dan kini seiring dengan kemajuan teknologi jaringan serta perkembangan dari teknologi digital, steganografi banyak dimanfaatkan untuk mengirim pesan melalui jaringan Internet tanpa diketahui orang lain dengan menggunakan media digital berupa file citra.

Tujuan dari steganografi adalah untuk mengirimkan suatu pesan melalui beberapa media baik berupa teks, gambar, audio dan video melalui kanal komunikasi di mana keberadaan dari pesan itu dirahasiakan. Berdasarkan pada Gambar 1, steganografi adalah salah satu dari teknik-teknik penyembunyian informasi dan yang dapat yang digolongkan ke dalam *linguistic steganography* dan *technical steganography* [4].



Gambar 1. Klasifikasi dari teknik penyembunyian informasi [4]

Linguistic steganography didefinisikan oleh Chapman et al. [2] merupakan "seni menggunakan bahasa alami yang ditulis untuk merahasiakan pesan-pesan rahasia". Namun pada saat ini teknik steganografi yang digunakan sudah amat beragam, beragam mulai dari algoritma yang digunakan sampai pada media yang digunakan. Beberapa contoh media penyisipan pesan rahasia yang digunakan dalam teknik Steganography antara lain adalah [1]:

1. Teks
   Dalam algoritma Steganography yang menggunakan teks sebagai media penyisipannya biasanya digunakan teknik NLP sehingga teks yang telah disisipi pesan rahasia tidak akan mencurigakan untuk orang yang melihatnya.
2. Audio
   Format ini pun sering dipilih karena biasanya berkas dengan format ini berukuran relatif besar. Sehingga dapat menampung pesan rahasia dalam jumlah yang besar pula.
3. Citra
   Format pun paling sering digunakan, karena format ini merupakan salah satu format file yang sering dipertukarkan dalam dunia internet. Alasan lainnya adalah banyaknya tersedia algoritma Steganography untuk media penampung yang berupa citra.
4. Video
   Format ini memang merupakan format dengan ukuran file yang relatif sangat besar namun jarang digunakan karena ukurannya yang terlalu besar sehingga mengurangi kepraktisannya dan juga kurangnya algoritma yang mendukung format ini.

Steganografi berfungsi untuk menyembunyikan keberadaan pesan dan dapat dianggap sebagai pelengkap dari kriptografi yang bertujuan untuk menyembunyikan isi pesan. Berbeda dengan kriptografi dalam steganografi pesan disembunyikan sedemikian rupa sehingga pihak lain tidak dapat mengetahui adanya pesan rahasia. Pesan rahasia tidak diubah menjadi karakter 'aneh' seperti halnya kriptografi. Pesan tersebut hanya disembunyikan ke dalam suatu media berupa gambar, teks, musik, atau media digital lainnya dan terlihat seperti pesan biasa.

Untuk memudahkan dalam proses penyembunyian pesan teks ke dalam *image*, maka dirancang suatu aplikasi steganografi dengan metode *Least Significant Bit* (LSB). Aplikasi dirancang dengan tiga proses yaitu mengambil *image*, menambahkan pesan ke dalam *image (encode image)* dan menampilkan pesan rahasia *(extract)* dalam *image*.

2. PERMASALAHAN

Steganografi mempunyai kelebihan dalam aspek penyembunyian pesan di mana pesan yang disembunyikan tidak terlihat kasat mata berupa kode tertentu seperti kriptografi, dikarenakan dalam steganografi pesan dititipkan pada suatu cover image. Permasalahannya bagaimana agar pesan tersebut dapat dititipkan pada *cover image* tanpa terlihat berkurangnya kualitas dari *cover image* tersebut, dan metode apa yang tepat agar pesan yang dititipkan tidak mengurangi kualitas *cover image*. Permasalahan berikutnya yaitu bagaimana merancang suatu *Graphical User Interface* (GUI) yang mudah digunakan oleh seorang yang awam sekalipun dengan steganografi, GUI harus dapat menampilkan *image*, teks yang dikirim, teks yang di *extract* dan menampilkan tiga menu dan satu tombol eksekusi yaitu *open image*, *create encode image*, *save image* dan *get massage*.

3. PEMBAHASAN

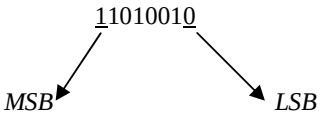
Steganografi yang dibahas di sini adalah penyembunyian data di dalam citra digital. Meskipun demikian, penyembunyian data dapat juga dilakukan pada wadah berupa suara digital, teks, ataupun video. Penyembunyian data rahasia ke dalam citra digital akan mengubah kualitas citra tersebut. Kriteria yang harus diperhatikan dalam penyembunyian data adalah:

1. Mutu citra penampung tidak jauh berubah. Setelah penambahan data rahasia, citra hasil steganografi masih terlihat dengan baik. Pengamat tidak mengetahui kalau di dalam citra tersebut terdapat data rahasia.
2. Data yang disembunyikan harus mampu bertahan terhadap manipulasi yang dilakukan pada citra penampung. Bila pada citra dilakukan operasi pengolahan citra, maka data yang disembunyikan tidak rusak.
3. Data yang disembunyikan harus dapat diungkapkan kembali (*recovery*).

Data yang dijadikan *cover* harus lebih besar dari data rahasia agar data rahasia tidak terlihat. Data yang digunakan sebagai *cover* sebaiknya digunakan satu kali. Apabila digunakan lebih dari satu kali, maka akan menimbulkan kecurigaan pihak lain [5].

3.1 Teknik Penyembunyian Data

Penyembunyian data dilakukan dengan mengganti bit-bit data di dalam segmen citra dengan bit-bit data rahasia. Salah satu metode penyembunyian data yang sederhana adalah *LSB Modification*.
Perhatikan contoh sebuah susunan bit pada sebuah *byte*:



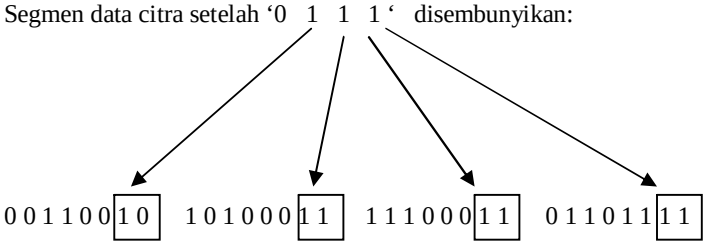
LSB = Least Significant Bit
MSB = Most Siginificant Bit

Bit yang cocok untuk diganti adalah bit *LSB*, sebab perubahan tersebut hanya mengubah nilai *byte* satu lebih tinggi atau satu lebih rendah dari nilai sebelumnya. Misalkan *byte* tersebut menyatakan warna keabuan tertentu, maka perubahan satu bit LSB tidak mengubah warna keabuan tersebut secara berarti. Lagi pula, mata manusia tidak dapat membedakan perubahan yang kecil.

Misalkan segmen data citra sebelum perubahan [3]:

0 0 1 1 0 0 1 1 1 0 1 0 0 0 1 0 1 1 1 0 0 0 1 0 0 1 1 0 1 1 1 1

Segmen data citra setelah ‘0 1 1 1 ‘ disembunyikan:



Untuk memperkuat teknik penyembunyian data, bit-bit data rahasia tidak digunakan mengganti *byte-byte* yang berurutan, namun dipilih susunan *byte* secara acak. Misalnya jika terdapat 50 *byte* dan 6 bit data yang akan disembunyikan, maka maka *byte* yang diganti bit *LSB*-nya dipilih secara acak, misalkan *byte* nomor 36, 5, 21, 10, 18, 49. Bilangan acak dibangkitkan dengan *pseudo-random-number-generator* (*PRNG*) kriptografi. *PRNG* kriptografi sebenarnya adalah algoritma kriptografi yang digunakan untuk enkripsi. *PRNG* dibangun dengan algoritma *DES* (*Data Encryption Standard*), algoritma *hash MD5*, dan mode kriptografi *CFB* (*Chiper-Feedback Mode*). Tujuan dari enkripsi adalah menghasilkan sekumpulan bilangan acak yang sama untuk setiap kunci enkripsi yang sama. Bilangan acak dihasilkan dengan cara memilih bit-bit dari sebuah blok data hasil enkripsi. Teknik penyembunyian data untuk citra 8-bit berbeda dengan citra 24-bit. Seperti diketahui berkas citra *bitmap* terdiri atas bagian *header*, palet *RGB*, dan data *bitmap.*

Pada citra 8-bit, setiap elemen data *bitmap* menyatakan indeks dari peta warnanya di palet *RGB.*

Format citra 8-bit (256 warna)

<*header*>

<palet *RGB*>

	R	*G*	*B*
1	20	45	24
2	14	13	16
3	12	17	15
…			
256	46	78	25

<data *bitmap*>
2 2 1 1 1 3 5 …

.

Pada citra 24-bit, tidak terdapat palet *RGB*, karena nilai *RGB* langsung diuraikan dalam data *bitmap*. Setiap elemen data *bitmap* panjangnya 3 *byte*, masing-masing *byte* menyatakan komponen *R*, *G*, dan *B*.

Format citra 24-bit (16 juta warna)

<*header*>
<data *bitmap*>
2 2 1 1 1 3 5 …

Pada contoh format citra 24-bit di atas, *pixel* pertama mempunyai $R = 2$, $G = 2$, $B = 1$.

3.2 Teknik Penggantian Bit pada Citra Bukan 24-Bit.

Sebelum melakukan penggantian bit *LSB*, semua data citra yang bukan tipe 24-bit diubah menjadi format 24-bit. Jadi, setiap data *pixel* sudah mengandung komponen *RGB*. Setiap *byte* di dalam data *bitmap* diganti satu bit *LSB*-nya dengan bit data yang akan disembunyikan. Jika *byte* tersebut merupakan komponen hijau (G), maka penggantian 1 bit LSB-nya hanya mengubah sedikit kadar warna hijau, dan perubahan ini tidak terdeteksi oleh mata manusia.

3.3 Teknik Penggantian Bit pada Citra 24-Bit.

Karena data bitmap pada citra 24-bit sudah tersusun atas komponen *RGB*, maka tidak perlu dilakukan perubahan format.

Setiap *byte* di dalam data bitmap diganti satu bit *LSB*-nya dengan bit data yang akan disembunyikan.

3.4 Teknik Pengungkapan Data

Data yang disembunyikan di dalam citra dapat dibaca kembali dengan cara pengungkapan (*reveal* atau *extraction*). Posisi *byte* yang menyimpan bit data dapat diketahui dari bilangan acak yang dibangkitkan oleh *PRNG*. Karena algoritma kriptografi yang digunakan menggunakan kunci pada proses enkripsi, maka kunci yang sama digunakan untuk membangkitkan bilangan acak. Bilangan acak yang dihasilkan sama dengan bilangan acak yang dipakai pada waktu penyembunyian data. Dengan demikian, bit-bit data rahasia yang bertaburan di dalam citra dapat dikumpulkan kembali.

3.5 Penyisipan Pesan ke dalam *Image*

Dengan menggunakan bahasa pemrograman java SE, fungsi yang akan digunakan untuk *encoding* atau menyisipkan pesan ke dalam *image* :

```
private static int[][][] encodeMessage(int[][][] origData, int cols, int rows,
String msg)
{
    int[][][] imgData = new int[rows][cols][4];1
    for (int row = 0;row < rows;row++) {
        for (int col = 0;col < cols;col++) {
            imgData[row][col][0] = origData[row][col][0];
            imgData[row][col][1] = origData[row][col][1];
            imgData[row][col][2] = origData[row][col][2];
            imgData[row][col][3] = origData[row][col][3];
        }
    }
    byte[] msgBytes = null;
    msgBytes = msg.getBytes("ISO-8859-1");2

    if ((msgBytes.length + 1 % 3 != 0)) {3
        int toAdd = (3 - (msgBytes.length % 3)) - 1;
        byte tmpBytes[] = new byte[msgBytes.length + toAdd];

        for (int i = 0; i < toAdd; i++) {
            tmpBytes[msgBytes.length + i] = (byte)'!';
        }
        System.arraycopy(msgBytes, 0, tmpBytes, 0, msgBytes.length);
        msgBytes = tmpBytes;
```

Gambar 2. *Encoding* Pesan

 ISBN 979 - 26 - 0255 - 0

3.6 Extract Pesan

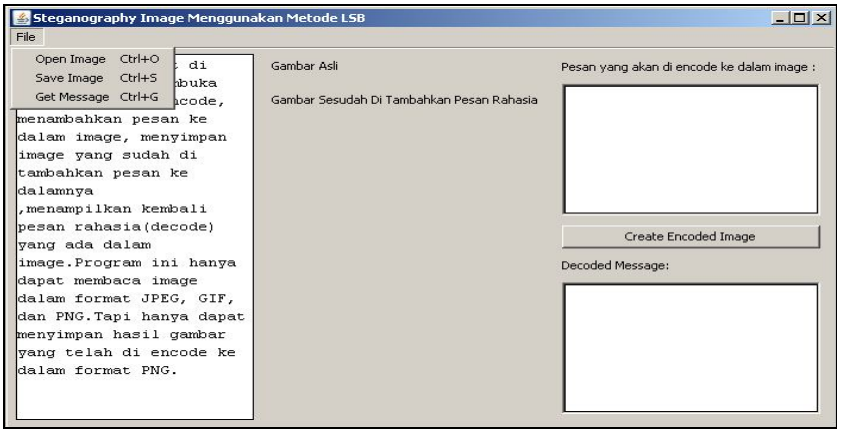
Fungsi yang akan di gunakan untuk mendecode image yang telah di tambahkan pesan rahasia ke dalamnya :

```
private static String getMessage(int[][][] data, int cols, int rows)
{
    int skipCount = 0;1
    int twoBitCount = 0;
    m_startCharCount = 0;
    m_foundTerminator = false;2
    byte[] twoBitData = new byte[(rows * cols - INSERTIONPOINT) * 3];3
    int byteCount = 0;
    StringBuffer message = new StringBuffer();
    for (int row=0; row < rows; row++) {
        for (int col=0; col < cols; col++) {
            if ((row * col > INSERTIONPOINT) && (skipCount-- == 0)) {
                twoBitData[twoBitCount++] = (byte) (data[row][col][1] & LSB_MASK_READ);4
                twoBitData[twoBitCount++] = (byte) (data[row][col][2] & LSB_MASK_READ);
                twoBitData[twoBitCount++] = (byte) (data[row][col][3] & LSB_MASK_READ);
                byteCount += 3;
                skipCount = twoBitData[twoBitCount - 1];5

                if ((byteCount % BUFFER_LENGTH) == 0) {6
                    message.append(decodeString(twoBitData,
                        twoBitCount - BUFFER_LENGTH, twoBitCount));
                    if (m_foundTerminator) {7
                        return message.toString();
                    }
                    }
                }
            }
     return null;
}
```

Gambar 3. *Extract* Pesan

Tampilan awal dari program utama – aplikasi steganografi untuk menyisipkan pesan ke dalam image :



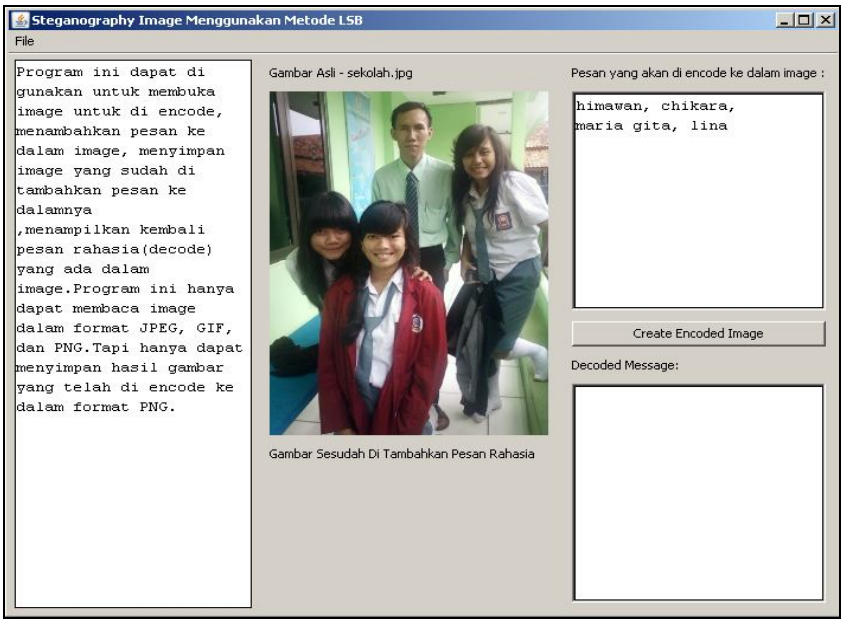
Gambar 4. Tampilan Awal Aplikasi Steganografi

3.7 Langkah-Langkah Proses Penyisipan Pesan ke Dalam File Image

Pertama user harus memilih image yang akan disisipkan pesan rahasia ke dalamnya dengan memilih menu *open image*, setelah user meload image yang ingin di tambahkan pesan ke dalamnya maka user bisa menuliskan pesan rahasia pada

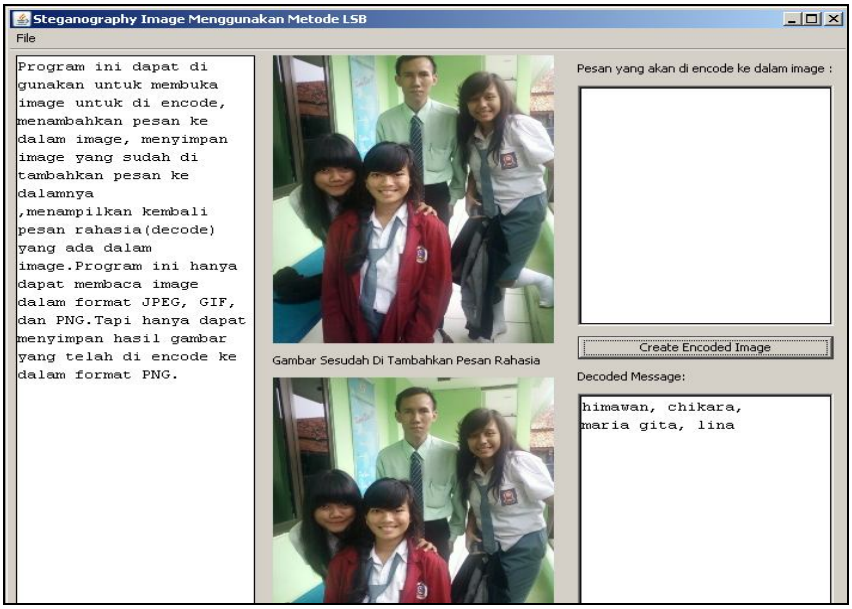
 ISBN 979 - 26 - 0255 - 0

textarea yang telah di sediakan pada sisi kanan frame aplikasi steganografi dan menekan button "***create encoded image***". Berikut adalah contoh bagaimana *image* yang telah di *load* dan ingin di tambahkan pesan ke dalamnya :
Pada contoh di bawah ini, resolusi (ukuran) dari image yang di gunakan sebesar 235 x 314 pixel dan format *image* yang di gunakan adalah .jpg.



Gambar 5. *Cover image* dan teks yang akan disembunyikan

Hasil tampilan *output image* yang telah disisipkan pesan rahasia tidak akan mengalami perubahan kualitas *image* yang signifikan seperti kualitas warna *(brightness atau contrast)*, berikut adalah tampilan output program dari hasil image yang telah di tambahkan pesan rahasia di dalamnya :



Gambar 6. *Extract encoding*

Hasil *extract*, dapat dilihat pada *decode message*, untuk ukuran (size) dari file image yang telah di sisipkan pesan rahasia menjadi bertambah besar. Ukuran semula pada image original dengan format .jpg = 34.7 kb, ukuran *image* setelah di tambahkan pesan rahasia dengan menggunakan format .png = 185 kb.

3.8 Running Aplikasi Steganografi :

Aplikasi steganografi yang dibuat ini terdiri dari 3 proses yaitu : me*load image* yang ingin ditambahkan pesan rahasia, menambahkan pesan ke dalam *image (encode image)* dan proses yang digunakan untuk menampilkan *(extract)* pesan rahasia yang ada dalam *image*. Proses me*load image* sangat sederhana, user memilih menu *"Open Image"* yang ada dalam aplikasi untuk mencari image yang ingin disisipkan pesan. Setelah image berhasil di*load* dan ditampilkan dalam aplikasi, maka user bisa menambahkan pesan rahasia ke dalam image kedalam text area yang telah di sediakan. Setelah pesan yang di tambahkan selesai di ketik, maka user menekan button *"Create Encoded Image"* untuk menambahkan pesan ke dalam image. Untuk proses menampilkan pesan rahasia yang ada dalam image, maka user bisa langsung memilih menu *"Get Message"* untuk mendapatkan pesan tersembunyi dalam *image*.

4. SIMPULAN

Aplikasi steganografi ini sangat bermanfaat terutama dalam perlindungan hak cipta *(copyright)* sebuah *image* dan juga keamanan pada saat melakukan transfer/pengiriman data melalui jaringan internet. Ukuran semula pada *image original* dengan format .jpg akan mengalami kenaikan nilai ditambahkan pesan rahasia dengan menggunakan format .png. Aplikasi steganografi sudah berpenampilan GUI mempunyai tiga menu yaitu *open image, save image, get message* dan satu tombol eksekusi yaitu *create encode image*. Hasil ekstrak image baik berupa teks, image, video dan audio akan terbuka begitu saja, sehingga untuk keamanan yang lebih baik maka disarankan dienkripsi kembali hasil *extract* tersebut.

DAFTAR PUSTAKA

[1] Alatas Putri, M. Subali,"Implementation Technique With Steganography LSB Method in Digital Images", Undergraduate Program, Faculty of Computer Science, Gunadarma University, 2009
[2] Chapman M, G. Davida, and M. Rennhard, "A Practical and Effective Approach to Large-Scale Automated Linguistic Steganography", *Proceedings of the Information Security Conference*, October 2001, pp. 156-165.
[3] Gusmayuda Rizki A,"Steganografi pada Media Video Digital dengan Menggunakan Metode FFT(Fast Fourier Transform) dan LSB (Least Significant Bit), Teknik Informatika, Fakultas Teknik dan Ilmu Komputer, Universitas Komputer Indonesia
[4] L. Y. POR, B. Delina," Information Hiding: A New Approach in Text Steganography", on Applied Computer & Applied Computational Science (ACACOS '08), Hangzhou, China, April 6-8, 2008
[5] Riko Arlando Saragih,"Metode Parity Coding Versus Metode Spread Spectrum pada Audio Steganography, Seminar Nasional Aplikasi Teknologi Informasi 2006 (SNATI 2006)